COLIBRI: SMT solving with CP

Bruno Marre Franck Vedrine François Bobot
Christophe Junke

May 23, 2025

COLIBRI[1] is a library (COnstraint LIBrary for veRIfication) developed at CEA LIST and used for verification or test data generation purposes since 2000, using the techniques of constraint programming. The variety of types and constraints provided by COLIBRI makes it possible to use it in many testing tools at CEA LIST like GATeL [2], for model based testing and verification from Lustre/SCADE, and Osmose [1], for structural testing from binary code, PathCrawler tool for concolic C testing.

COLIBRI is a finite domain solver that use usual constraint programming techniques. The basis is that one or many domains are attached to each terms, and the constraints tighten the domains of one term using the domains of the other terms involved in the constraint. When no constraints could improve anymore the domain of any term, a decision is made. Usually it splits the domain in two, but other techniques can be used as 3B filtering which decide on the extremes of the domains and if unsatisfiable try to improve this bound.

The domains are very specific, no reduction to simpler theory like bit-blasting is used. COLIBRI uses a domain of union of intervals for real and integer inter-reduced with a domain of congruence. For real domain, it may additionaly try some instantiations with rationnal values. For floating points it uses a domain of intervals. For bit-vector a domain that indicates if the bits are set, unset or unknown.

In addition to these domains which reason on local properties, COLIBRI uses DBM for bit-vectors, integers, reals, and floating points. The DBM uses lots of patterns to be able to do limited but useful non-linear reasoning. The power of the reasoning of COLIBRI, in particular in the floating points, are due to the information sharing and inter dependencies of all these reasoning techniques.

The combination of all the components of COLIBRI is simplified since, like all CP-solver, all the domains and constraints are improving the same model.

Some improvements were made for the NRA theory in order to recognize (square) remarkable identities. This allow to infer the sign of some expressions and to handle more accurate constraints for their reductions. These improvements can be extended to other power of remarkable identities. Moreover, an experimental extension of reals operators was proposed for the handling of transcendental functions: colibri_$(sin|asin)$, colibri_$(cos|acos)$ and colibri_$(tan|atan)$ for trigonometry, and colibri_$(ln|exp)$ for other common transcendental functions. For these extensions, labeling with rational numbers is

[1] http://colibri.frama-c.com

not allowed. Recently a partial support of quantifiers has been added : sat positive “forall” as well as unsat negative “exists” are not reliable and return unknown.

References

[1] S. Bardin and P. Herrmann. Structural testing of executables. In *the First International Conference on Software Testing, Verification, and Validation (ICST 2008)*, pages 22–31. IEEE Computer Society, 2008.

[2] B. Marre and B. Blanc. Test selection strategies for Lustre descriptions in GATeL. *Electronic Notes in Theoretical Computer Science*, 111:93–111, 2005.